FMS: the GFDL Flexible Modeling System

V. Balaji
SGI/GFDL Princeton University

NOAATech 2002
Silver Spring MD
25 October 2001

GFDL Strategic Objectives

GFDL is a NOAA climate modeling centre. The primary focus is the use and development of coupled climate models for simulations of climate variability and climate change on short (seasonal-interannual) and long (decadal-centennial) time scales.

- Provide timely and reliable knowledge for the nation on natural climate variability and anthropogenic change.
- Develop Earth Systems Models (ESMs) for climate variability and change.
- Advance expert assessment of global and regional climate change through research, improved model and data products.

GFDL Computing

- Reliance on Cray vector architecture in previous decades.
- Transition to scalable computing begun in 1997 with the acquisition of Cray T3E.
- Current computing capability: $2 \times 256 + 6 \times 128 + 2 \times 64$p Origin 3000.

Technological trends

In climate research... increased emphasis on detailed representation of individual physical processes governing the climate; requires many teams of specialists to be able to contribute components to an overall coupled system;

In computing technology... increase in hardware and software complexity in high-performance computing, as we shift toward the use of scalable computing architectures.

The GFDL response:

modernization of modeling software

- Abstraction of underlying hardware to provide *uniform programming model* across vector, uniprocessor and scalable architectures;

- Distributed development model: many contributing authors. Use high-level abstract language features to facilitate development process;

- Modular design for interchangeable dynamical cores and physical parameterizations, development of *community-wide standards* for components.

FMS: the GFDL Flexible Modeling System

Jeff Anderson, V. Balaji, Matt Harrison, Isaac Held, Paul Kushner, Ron Pacanowski, Pete Phillipps, Bruce Wyman, ...

- Develop high-performance kernels for the numerical algorithms underlying non-linear flow and physical processes in complex fluids;
- Maintain high-level code structure needed to harness component models and representations of climate subsystems developed by independent groups of researchers;
- Establish standards, and provide a shared software infrastructure implementing those standards, for the construction of climate models and model components portable across a variety of scalable architectures.
- Benchmarked on a wide variety of high-end computing systems;
- Run in production on very different architectures: parallel vector (PVP), distributed massively-parallel (MPP) and distributed shared-memory (NUMA).

FMS design principles

Modularity data-hiding, encapsulation, self-sufficiency;

Portability adherence to official language standards, the use of community-standard software packages, compliance with internal standards;

Flexibility address a wide variety of climate issues by configuring particular experiments out of a wide choice of available components and modules.

Extensibility attempt to anticipate future needs: choices for the same physical function to present similar external interfaces;

Community users encouraged to become developers by contributing components: public release of infrastructure, components and complete model configurations.

Architecture of FMS

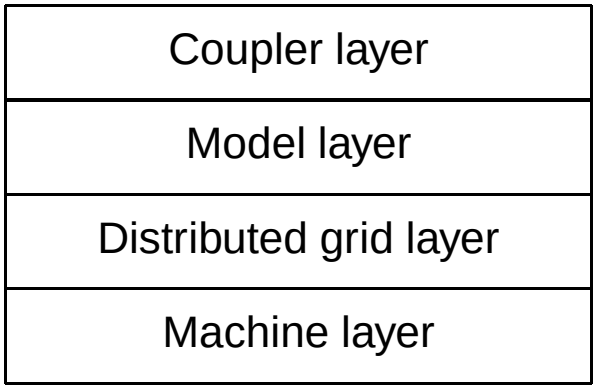
Coupler layer scheduling of component models, data exchange between component models.

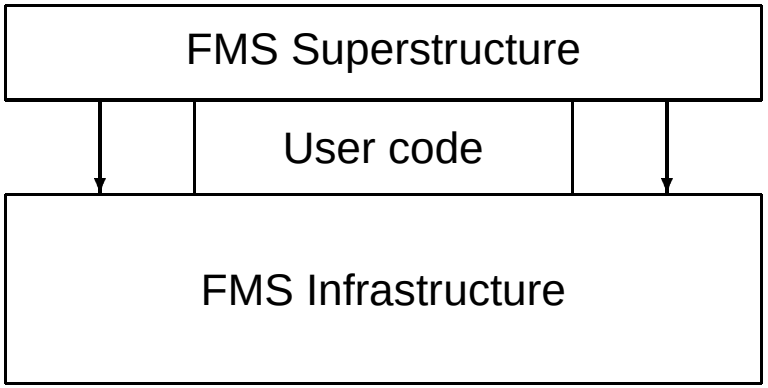
Model layer component models (atmosphere, ocean, etc.) compliant with the framework code standards.

Distributed grid layer standard for physical description of model fields on spatial grids distributed across parallel systems, and parallel operations on these fields.

Machine layer communication primitives (MPI, shmem, etc), I/O, other platform-specific operations.

Architecture of FMS





FMS shared infrastructure:

machine and grid layers

MPP modules communication kernels, domain decomposition and update, parallel I/O.

Time and calendar manager tracking of model time, scheduling of events based on model time.

Diagnostics manager Runtime output of model fields.

Scientific libraries Uniform interface to proprietary and open scientific library routines.

Communication kernels

provide uniform interface to:

- MPI message-passing across clusters.
- MPI or SHMEM on tightly-coupled distributed memory (T3E).
- Pointer-sharing and direct copy on shared-memory and distributed-shared memory (NUMA).

Coupler

Used for data exchange between models. Key features include:

Conservation: required for long runs.

Resolution: no constraints on component model timesteps and spatial grid. Supports both explicit and implicit timestepping.

Exchange grid: union of component model grids, where detailed flux computations are performed (Monin-Obukhov, tridiagonal solver for implicit diffusion, ...)

Fully parallel: Calls are entirely processor-local: exchange software will perform all inter-processor communication.

Modular design: uniform interface to main calling program.

No brokering: each experiment must explicitly set up field pairs.

Single executable.

FMS component models

- Atmosphere:
  - BGRID: hydrostatic finite difference model on a staggered Arakawa B grid and hybrid σ/P vertical coordinate (Wyman);
  - SPECTRAL: hydrostatic spectral transform model also with the hybrid σ/P vertical coordinate (Held, Phillipps);
- Ocean: MOM primitive equation ocean climate model with generalized horizontal coordinates and vertical z-coordinate, full suite of physics options, compatible with state-of-art adjoint compiler (Pacanowski, Griffies, Rosati, Harrison);
- Ice: Sea Ice Simulator (SIS) full sea ice dynamics with elastic-plastic-viscous rheology, N-category ice thickness, 3-layer vertical thermodynamics (Winton);
- Land: Land Dynamics model (LaD) 5 temperature layers, 11 soil/vegetation types, stomatal resistance, bucket hydrology, river routing (Milly);

Atmospheric physics options

Radiation diurnal cycle, radiative effects of trace gases; Simplified Exchange Approximation (LW), liquid/ice cloud radiative properties (SW);

Convection Relaxed Arakawa-Schubert, convective-stratiform detrainment;

Clouds 3 prognostic tracers, prognostic stratiform clouds;

PBL Mellor-Yamada 2.5, Monin-Obukhov similarity theory;

Gravity wave drag Pierrehumbert-Stern.

Current GFDL activities using FMS

- Tuning of atmospheric and coupled models for radiative balance for decadal-centennial simulations;
- Development of seasonal-interannual forecasting capabilities;
- Mesoscale eddy-permitting simulations of the southern ocean;
- Incorporation of global biogeochemical models into coupled model for carbon cycle modeling.

Future developments: algorithms and models

- New atmospheric physics options currently under testing:
  - Donner deep convection including convective updrafts and MCCs, convective and mesoscale downdrafts;
  - Bretherton-Grenier PBL;
  - Held-Klein very stable PBL;
  - Enhances convective (Alexander-Dunkerton) and mountain gravity wave drag, inclusion of stratosphere in model.
- LaD model architecture to permit modular subsystems for vegetation and soil hydrology; incorporating ED (ecosystem demography) and VIC (Variable Infiltration Capacity) models from Princeton for dynamic vegetation and hydrology data assimilation capacities.
- Development of non-hydrostatic atmospheric model options.

Future developments: public release

A staged public release of FMS is currently underway:

- infrastructure
- components and modules
- complete model configurations.

These stages will take place between December 2001 and September 2002.

Future developments: GFDL and community standards

- FMS authors are now leading participants in the design of the Earth Systems Modeling Framework (ESMF) community-wide modeling standard and framework, for which FMS is a design prototype. ESMF is curreently scheduled for public release between 2003 and 2005.

- Work with OAR and NWS to put all key NOAA models on community standard infrastructure;

- Establish working partnerships with non-NOAA community (NCAR, NASA, universities); joint workshops and other activities to resolve specific scientific and modeling issues.

Domain decomposition and domain update

- mpp_define_domains()
- mpp_update_domains()

```
type(domain2D) :: domain(0:npes-1)
call mpp_define_domains( (/1,ni,1,nj/), domain, xhalo=2, yhalo=2 )
...
!allocate f(i,j) on data domain
!compute f(i,j) on compute domain
...
call mpp_update_domains( f, domain(pe) )
```

Parallel I/O

```
type(domain2D) :: domain(0:npes-1)
type(axistype) :: x, y, z, t
type(fieldtype) :: field
integer :: unit
character*(*) :: file
real, allocatable :: f(:,:,:)
call mpp_define_domains( (/1,ni,1,nj/), domain )
call mpp_open( unit, file, action=MPP_WRONLY, format=MPP_IEEE32, &
  access=MPP_SEQUENTIAL, threading=MPP_MULTI, fileset=MPP_MULTI )
call mpp_write_meta( unit, x, 'X', 'km', ... )
...
call mpp_write_meta( unit, field, (/x,y,z,t/), 'Temperature', 'kelvin', ... )
...
call mpp_write( unit, field, domain(pe), f, tstamp )
```

Parallel I/O output modes

Three types of parallel I/O:

- Single-threaded I/O: a single PE acquires all the data and writes it out.
- Multi-threaded, single-fileset I/O: many PEs write to a single file.
- Multi-threaded, multi-fileset (distributed) I/O: many PEs write to independent files (requires post-processing).